Knauf, Rainer; Sakurai, Yoshitaka ; Tsuruta, Setsuo:

Applying knowledge engineering methods to didactic knowledge: first steps towards an ultimate goal

Zuerst erschienen in:
IEEE International Joint Conference on Neural Networks, 2008 : IJCNN 2008 ; [part of] IEEE World Congress on Computational Intelligence [(WCCI 2008)] ; 1 - 8 June 2008, [Hong Kong, China]. - Piscataway, NJ : IEEE, 2008. - ISBN 978-1-4244-1820-6. - S. 38-45
DOI: 10.1109/IJCNN.2008.4633764

Applying Knowledge Engineering Methods to Didactic Knowledge First Steps Towards an Ultimate Goal

Rainer Knauf, Yoshitaka Sakurai and Setsuo Tsuruta

Abstract— **Generally, learning systems suffer from a lack of an explicit and adaptable didactic design. Since E-Learning systems are digital by their very nature, their introduction rises the issue of modeling the didactic design in a way that implies the chance to apply Knowledge Engineering Techniques (like Machine Learning and Data Mining). A modeling approach called storyboarding, is outlined here. Storyboarding is setting the stage to apply Knowledge Engineering Technologies to verify and validate the didactics behind a learning process. Moreover, didactics can be refined according to revealed weaknesses and proven excellence and successful didactic patterns can be inductively inferred by analyzing the particular knowledge processing and its alleged contribution to learning success.**

I. Introduction

The design of learning activities in collegiate instruction is a very interdisciplinary process. Besides deep, topical knowledge in the subject being taught, an instructor needs knowledge and skills in many other subjects. This includes IT-related skills to use today's presentation equipment, didactic skills to effectively present the topical content, plus skills in fields like social sciences, psychology and ergonomics.

In particular, university instruction often suffers from a lack of didactic design. Since universities are also research institutions, their professors are usually hired based on their topical skills. Didactic skills are often underestimated in the recruiting process.

Our approach to facing problems like these is a modeling concept for didactic knowledge called Storyboarding. A storyboard provides a roadmap for a course, including alternative paths and possible detours if certain concepts to be learned need reinforcement. Using modern media technology, a story-board also plays the role of a server that provides the appropriate content material when deemed required. Our suggestion to ensure a wide dissemination of this concept is to use a standard tool to develop and process this model, which is Microsoft Visio. Additionally, we developed a platform independent web based storyboard development environment [14], which allows the design of storyboards while guaranteeing their logical soundness.

The paper is organized as follows. Section 2 is an introduction to the storyboard concept. In includes the present state of the current development. Section 3 gives an overview on Knowledge Engineering Technologies, which have been developed and implemented for storyboards. In section 4, we summarize the research undertaken so far and outline current work as well as research horizons.

Rainer Knauf is with the Faculty of Computer Science and Automation, University of Ilmenau, PO Box 100565, 98684 Ilmenau, Germany (phone: +49 (0)3677 69 1445; fax: +49 (0)3677 69 1665; email: rainer.knauf@tu-ilmenau.de). Yoshitaka Sakurai and Setsuo Tsuruta are with the School of Information Enviroment at Tokoy Denki University.

II. Storyboarding

A. Former Storyboarding Concepts

Former Storyboarding concepts to model information and learning processes have been introduced 1998/1999 [10]. The employment of storyboarding approaches for (unfortunately, only) e-learning is characterized by misunderstandings. So-called storyboard concepts in use are mostly substitutes for software-technological documents of high-level design, but are not very much specific to the instructional design process [4], [18]. Didactic concepts [11] are not made explicit and, thus, pondering about didactics is not sufficiently enforced. Again, also very recent approaches as introduced above (see also [17]) remain within the borders of IT systems.

There are contrasting approaches [12] that are conceptually very useful, but syntactically much too far from a workflow directed to technology enhanced learning implementations. The crux is that purely software-technologically driven concepts do not provide an opportunity to represent and discuss details of human learning [3] [5]. Learning is much more than memorizing: "Learning imposes new patterns of organization on the brain, and this phenomenon has been confirmed by electrophysiological recordings of the activity of nerve cells." ([3], p. 121).

Learning is reasonably understood as an interactive knowledge construction process. Illustrative case studies are discussed in [6]. This book's chapter "3B Organizing Shapes" reports process of conversation and co-operation between a teacher and his students in which a variety of media types, forms of interaction, and learners' activities are dovetailed. Didactic design means the anticipation of those communication processes [11], and storyboards may provide the expressive power suitable to the design and implementation of learning processes. This, however, needs to go beyond the limits of software systems specification – the crucial question for innovations in didactic design.

B. Our Storyboarding Concept

Our storyboard concept is built upon standard concepts which enjoy (1) clarity by providing a high-level modeling approach, (2) simplicity, which enables everybody to become a storyboard author, and (3) visual appearance as graphs. With respect to a better formal composition, processing, verification, validation and refinement the concept as introduced so far [8] [13] has been further developed. We adopt these modifications. Here, we define a storyboard as follows:

978-1-4244-1821-3/08/$25.00

A storyboard is a nested hierarchy of directed graphs with annotated nodes and annotated edges. Nodes are scenes or episodes. Scenes denote leaves of the nesting hierarchy. Episodes denote a sub-graph. There is exactly one Start- and End- node to each (sub) graph. Edges specify transitions between nodes. They may be single-color or bi-color. Nodes and edges have (pre-defined) key attributes and may have free attributes.

The interpretations of these terms follow after presenting a small example.

The representation as a graph (instead of a linear sequence) reflects the fact that different readers trace the paper differently according to their particular interests, prerequisites, a current situation (like being under time pressure), and other circumstances. The story-board is the authors' design document representing expectations of human behavior. For

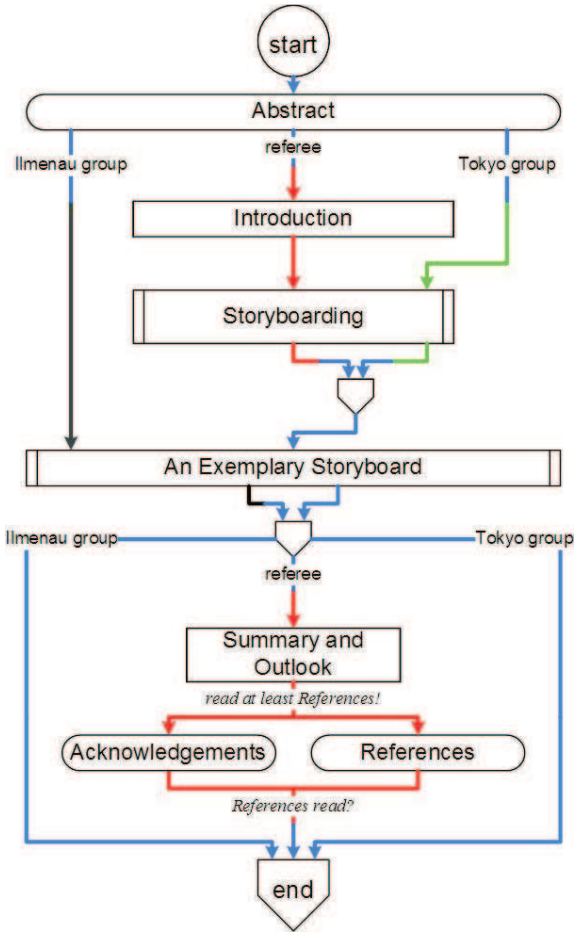


Fig. 1. A Storyboard on a Conference Paper

exemplification, Figure 1 shows a top level storyboard on one of the author's conference papers. Alternative paths may be driven by the reader's role:

- The Ilmenau research group may skip the Introduction, the section on storyboarding and the summary, acknowledgements and references, because they are familiar with it. Since the example application on higher level storyboards is new to them, they will study this example.
- The Tokyo research group may skip the introduction and the section on their so-called Dynamic Learning Need Reflection System (DLNRS) [7], [8]for similar reasons. In fact, the DLNRS is their original invention. However, they should read the section on storyboarding, because it contains some supplements that are new to them. Like the Ilmenau group, they are interested in studying the section on higher level storyboards and skip the rest of the paper.
- Referees (hopefully) want to read all. After the summary, they can read Acknowledgements and References independently in any sequence. For their duty they have to check the References at least.

A storyboard can be traversed in different manners according to (1) users' interests, objectives, and desires, (2) didactic preferences[1] , (3) the sequence of nodes (and other storyboards) visited before (i.e. according to the educational history), (4) available re-sources (like time, money, equipment to present mate-rial, and so on) and (5) other application driven circumstances. In fact, people may read the present paper in ways that are different from our assumptions modeled in Figure 1. However, for the ways we anticipate, we can ensure that they are coherent. A storyboard may be seen as a model of an anticipated reception process that is interpreted as follows:

- *Scenes* denote a non-decomposable learning activity that can be implemented in any way. It can be the presentation of a (media) document, opening a tool that supports learning (URL or e-learning system) or an informal activity description.
- *Episodes* are defined by their sub-graph.
- *Graphs are* interpreted by the paths, on which they can be *traversed*.
- A *Start Node* of a (sub-) graph defines the starting point of a legal graph traversing. An *End Node* of a (sub-) graph defines the final target point of a legal graph traversing.
- *Edges* denote transitions between nodes. There are rules to leave a node by an outgoing edge: (1) The outgoing edge must have the same color as the in-coming edge by which the node was reached. (2) If there is a condition specified as the edge's key at-tribute, this condition has to be met for leaving the node by this edge.
- *Key attributes of nodes* specify application driven information, which is necessary for all nodes of the same type, e.g. actors and locations. *Key attributes of edges* specify conditions, which have to be true for traversing on this edge.
- *Free attributes* specify whatever the storyboard author

[1] In the authors' experience, some students understand better by presenting illustrations, others by providing a small example and others by providing formal descriptions.

wants the user to know: didactic intentions, useful methods, necessary equipment, e.g.

The types of nodes and edges in our storyboard implementations are shown in tables I and II.

What are peculiarities of the concept? At a first view, this purpose is similar to the purpose of traditional storyboards that are produced for shows, plays, theater games or movies, i.e. visual arts. The materials and tools of the storyboarded learning activities (e.g., text books, scripts, slides, hard- and software models, e-learning systems and others) are something comparable to the requisites of a show. Basic differences of our storyboards to those used to "specify" a show are:

- the primary purpose (learning vs. entertainment),
- the degree of formalization, and, as a consequence of being semi-formal,
- the obligation of everything above the level of scenes, which does (and should) not apply to storyboards in arts, in which the intendant has some freedom of individual interpretation and
- (thanks to formalization) the opportunity to formally represent, process, evaluate, and refine our storyboards, which does not apply at all to story-boards in visual arts.

Also, Storyboards have somewhat in common with classic AI knowledge representations like *Semantic Networks* and *Frames* as well as with process modeling languages like *State Diagrams* and *Petri Nets* (see e.g. [1] for use in learning processes), *Workflow Diagrams* (see e.g. [16] for use in learning processes) and *Float Charts* (see e.g. [20] for use in learning processes). Items that make this concept more expressive for didactic knowledge than representations as mentioned above are

- the potentially unlimited nesting of graphs,
- the opportunity to express "conditioned" edges by using the colors (bi-colored edges, e.g.) or respective key annotations to edges,
- the opportunity to use (two kinds of) fork-edges,
- the potential of nodes to carry many different teaching materials and tools as hyperlinks[2], and, most important, and
- the fact that a scene can be implemented in any way, i.e. is not restricted to something electronically available or even formally structured (like any knowledge representations and any material included in process models).

III. Knowledge Engineering with Storyboards

A. Formal Verification of Storyboards

In fact, our concept of storyboarding is a semi-formal one. The graph hierachy is completely formal and below the level of *Scenes* is is completely informal. Thus, the *Scenes* form the interface between the formal and the informal levels.

[2]The author developed a storyboard for an AI course at an US university and included material of his own AI course in Germany. Now, this storyboard serves both universities and is also a common platform for internationally sharing teaching materials.

The formal levels increase the logical reliability such as consistency, completeness, and so on. To ensure consistency and completeness of our storyboards, we developed and implemented several verification procedures [19]:

1) An *Episode - Hierarchy - Test* focuses questions like:
   - Does every episode have exactly one related graph?
   - Does every (non-top) graph have exactly one related episode node in exactly one related super-graph?
2) Also, *reachability* issues are formally checked:
   - Is the End node reachable on every possible path in each (sub-) graph?
   - Is each node reachable from the Start node in each (sub-) graph?
3) Furthermore, *completeness* and *non-contradictoriness* of alternative outgoing edges (with the same beginning color) by logically analyzing conditions expressed as annotations of each node's outgoing with the same start color.
4) Edge colors, which express the *interdependence of incoming/outgoing edges*, are also a subject of formal verification by checking these issues:
   - Is there a unique (beginning) color of the Start node's outgoing edges?
   - Is there at least one outgoing edge with the same (beginning) color for each incoming edge's (finishing) colors?

The above mentioned anomaly tests are implemented for our storyboard implementations with Microsoft Visio^{TM} [9].

Episode Hierarchy Test

This test is twofold, top-down and bottom-up. First, every episode graph is checked, whether or not

- each graph, which is reachable from the top level graph, has exactly one related episode node of exactly one super graph and
- all episodes don't contain themselves directly or indirectly.

In fact, it may happen that a graph contains itself and thus, the reference chain forms an endless loop. Moreover, this may cause a non-reachability of other graphs. Figure 2 illustrates these cases. Here, Episode 1 contains itself and therefore, the episodes 1 and 2 refer each other without termination. Moreover, episode 3 can't be reached because of this circumstance. Passing this test is a proof that all graphs, which are reachable from the root (the to level storyboard) form a tree of graphs.

However, there also may be "lonely" episode nodes or graphs that need to be detected by this check as well. In Figure 3, both situations are illustrated. Here, episode 3 is a lonely episode node and episode 4 is a lonely episode graph.

Reachability Test

Detecting episodes with no incoming edge was subject of the previous test, because they indicate non-reachable (sub-)

TABLE I
NODE TYPES

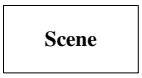
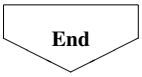
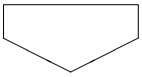
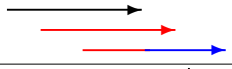
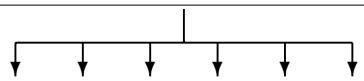
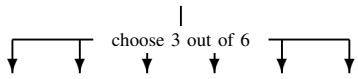
	Symbol	Behavior when following a hyperlink	Behavior when double clicked
Scene		• opening a document (*.doc, *.pdf, *.wav, *.vsd, *.ppt, *.xls , ...) with the appropriate tool • nothing, if just verbally described scene	• opening a document (*.doc, *.pdf, *.wav, *.vsd, *.ppt, *.xls , ...) with the appropriate tool • visiting a website with the standard browser in case it is an URL • opening the standard mail tool in case it is an e-mail address
Episode		opening a subgraph that specifies the episode	• opening a document (*.doc, *.pdf, *.wav, *.vsd, *.ppt, *.xls , ...) with the appropriate tool • visiting a website with the standard browser in case it is an URL • opening the standard mail tool in case it is an e-mail address
Start Node		jumping to the start node of the related super-graph	not meaningful
End Node		jumping to the *Reference Node* that successes it's associated *Episode* Node in the related super-graph	not meaningful
Reference Node		jumping to the *End Node* of the sub-graph that is associated to the preceded *Episode* Node	not meaningful

TABLE II
EDGE TYPES

	Symbol	Interpretation
Simple Edge		defines a unique successor node
Fork		defines several successor nodes, which have to be traversed independently from each other, i.e. in any sequence or parallel
Fork with conditions		defines several successor nodes, which have to be traversed independently from each other, i.e. in any sequence or parallel, according to the specified condition, e.g. take n out of m paths

graphs. However, also the other nodes need to be checked for reachability from the start node of their related graph. Furthermore, the end node of their related graph needs to be reachable from them. This is subject of the reachability test.

Color Tests

Bi-colored edges have a beginning color and change their color within the edge, i.e. they have an ending color that is different from the beginning color. Single-colored edges are considered as edges with the same beginning and ending color. Table II shows some examples for both. The color business serves to express the dependency of leaving a node from the way of entering it. This is necessary, if a node is used in different contexts, i.e. if the same node is part of different paths. For example, at an university, the same course may be taught to students of different subjects and the subsequent course depend on their subject and/or the courses they visited previously. Here, we check, whether or not for each beginning color of outgoing edges there is an incoming edge with the same ending color. If this condition is not met, the node is a "dead end". Since this is a consistency anomaly, this fact needs to be reported to the storyboard author. Vice versa, for each end color of an incoming edge there has to be an outgoing edge of the same beginning color. This is checked and reported, too.

Multiple Nodes / Multiple Edges

Since the color business allows setting a node into different contexts (see above), there is no need to represent nodes with identical content (subgraphs or scene-implementations) more than once. Also, there is no need for multiple edges.

Multiple edges are simple to identify. If various edges start

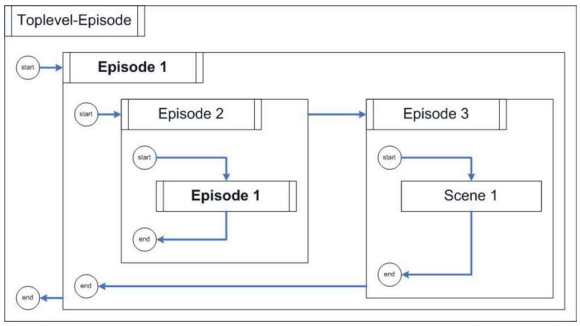


Fig. 2. Self-Containment and non-reachability caused by endless loops

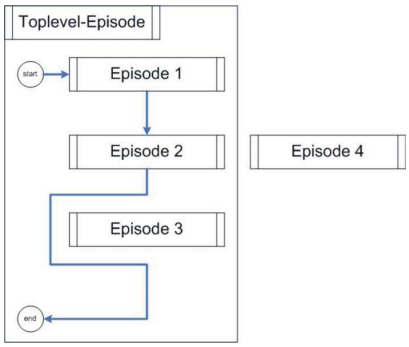


Fig. 3. Lonely nodes and graphs

and end at the same node and have the same beginning as well as ending color, these edges can be reduced to one edge.

For nodes, however, practice taught us, that authors, who develop huge storyboards over a long period of time don't remember all nodes implemented so far and may develop a node that already exists. Moreover, it happened that they name the new node slightly different from the existing (identical) one. Also, it happened that authors name nodes with different content identically.

Thus, we decided to detect nodes with identical identical or almost identical names and ask the author direct the author's attention to these nodes. We left it to the author to decide, whether these nodes are really identical (i.e. can be merged to one node) or just named (almost) identical accidentally. Of course, we can't tolerate complete identical names for different nodes. We addressed these problems by comparing node names in four different modes, namely

- *Binary mode* Here, nodes with identical names are identified.
- *Text mode* Here, the letters of the names are compared, but upper- and lower case are considered identical.
- *Metaphone mode*[3] Here, the node names are compared by their sound.
- *Soundex mode*[4] This is similar to the previous mode, but less sensitive when considering sounds as identical.

B. An Inheritance Concept

Additionally, an inheritance concept were implemented within the graph hierarchy, which distinguishes several inheritance types such as sum, maximum, or set union for inheritance within the graph hierarchy [21].

1) In some applications it makes sense to inherit annotations from nodes (both scenes and episodes) to their related super-graph. For example, material that are used to teach a particular lecture is also material to teach the complete subject the lecture is part of.
2) In other cases it makes sense to inherit the arithmetic sum of a key annotation of all nodes to the related super-graph. For example,
   - an upper limit of the time needed to teach a subject can be estimated by the sum of its components (lectures) and
   - maximum cost of a university study can be estimated by the sum of the fees for all recommended subjects.
3) In other cases it makes sense to inherit the maximum value of a key annotation of all odes to the related super-graph. For example, the educational difficulty (basic/easy, medium, advanced, very difficult) of a study needs to be communicated as the maximum value of all mandatory subjects.

Thus, for each key annotation an appropriate inheritance method can be selected in our Microsoft Visio ^{T}M implementation of storyboards.

From a Knowledge Engineering point of view, inheritance in the storyboard hierarchy is some sort of deductive inference over the knowledge represented as storyboards.

C. Towards a Storyboard Development Environment

For an a priori approach to ensure such logical features, a set of operations were defined [19], which's exclusive use automatically leads to a "legal storyboard". These operations are:

- Adding paths,
- Adding nodes,
- Turning Scenes to Episodes, which is a refinement of a scene by introducing a related sub-graph,
- Adding a concurrent path, and
- Merging (equivalent) nodes by introducing related bi-colored edges, which make sure that the linkage with the remaining graph isn't changed.

Since the last of these operations might be not easy to understand and imagine, it exemplarily illustrated in Figure 4. Here, V_1 and V_3 are equivalent and V_2 and V_4 are equivalent. Since different users visit them in different sequences, they are represented as different nodes on the left hand side. By

[3]Metaphone is a phonetic algorithm, for indexing words by their sound, when pronounced in English.

[4]Soundex is a phonetic algorithm for indexing names by sound, as pronounced in English. It is a little less sensitive than Metaphone with respect to considering sounds as identical.

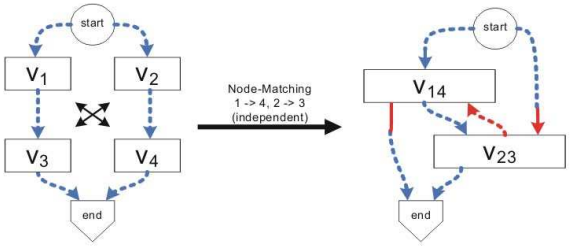


Fig. 4. Merging equivalent nodes

merging the equivalent nodes together, a new color needs to be introduced to express these different sequences.

Based on this set of operations we developed a web-based storyboard development environment for our storyboards [14], which is currently available at *http://sebastiank.awardspace.com*. By using this tool, the structure tests as listed up in section III-A are obsolete.

D. Data Mining over Storyboard Paths

A basic objective of this storyboard application is to use Knowledge Engineering technologies like *Data Mining* and *Case-Based Reasoning* on the (semi-) formal process models.

The objective of this research is inductively "learning" successful storyboard patterns and recommendable paths. This is some sort of meta-learning, i.e. the learning of learning knowledge. This is performed by an analysis of the paths where former students went through the storyboard and based on their success that is associated with these particular paths [2].

To more concretely show the feasibility and benefit of high level dynamic storyboarding for its qualified assistance of students suffering from the "jungle of opportunities and constraints" in university education, a simple prototype was recently developed to evaluate curricula created or modified by the students in advance of their study [2].

For this purpose, we introduced a concept to estimate success chances of curricula, which are composed by students at the School of Information Environment of the Tokyo Denki Uuiversity in their curriculum planning class in the first semester. Since the storyboard representation enjoys a certain degree of formality, there is an opportunity to apply data mining techniques on storyboarding paths that have been used by students. Furthermore, these paths can be associated with the student's related success, i.e. his/her final result of the study. Based on these examples, the success chance of intended paths can be estimated as follows. The concept is described in detail in [2]. Furthermore, [2] contains a prototypical implementation in Prolog, which shows its applicability.

Construction of a decision tree

The storyboard developed for TDU [7], [8] models the opportunities to form curricula. Here, the edges specify prerequisite conditions. The start node of an edge specifies a subject that is a prerequisite of the subject, at which the edge ends.

The construction of the decision tree is based on the paths of former students through the storyboards which model the "space of opportunities", in which the students took a particular one, which is a path through the storyboard. Each of those paths can be associated with the degree of success, which has been achieved by the student. In case a set of students went the same path, the degree of success can be estimated by a weighted average degree of them.

More concretely, this path starts at the start node of the top level storyboard and ends at its end node. For each episode on this path, the related episode is replaced by its sub-graph. This replacement is continued throughout the entire hierarchy of nested graphs.

Each scene of this storyboard application [7], [8] represents a subject students can enroll. Figuratively speaking, the decision tree is constructed on the basis of a "flatten" storyboard. Flatten, in this context, means the graph hierarchy it "flatten down" to just one level with no subgraph.

The implementation is realized in Prolog. The input is implicitly given by a database of predicates describing the storyboard. The storyboard is represented by Prolog facts

- *includes(graph, [<elements>])*
- *edge([<begin>], [<end>], [<(color, color)>])*

The edge-fact models the aspect of transition between nodes. They are, however, not used for decomposing. The method is based on the includes-predicates only and is performed as follows:

- First, we distinguish sequential from parallel structures. Sequential elements form a sub-path, parallel structures form a single element that is handled as one element of a path.
- Because the input is a list, we check each element, whether it is an episode by using the predicate includes.
- In case it is an episode, the elements of its sub-graph will be decomposed recursively.
- All atomic elements will be appended to the resulting path, which is the output of the method.

The decision tree is based on the concept of bundling common starting sequences [2] of the various paths to a knob of the tree. In [2] these starting sequences are called "least common denominator". Of course, all paths went by students start with the start node, which forms the root of the decision tree. Several first elements will result in several subtrees right below the root. This continues for each sub-tree accordingly, i.e. if different paths with a common starting sequence from the root until the actual node differ in their next (subsequent) node, related sub-trees will be established.

Each node in this tree, which represents a final node of a path, is followed by a label-node, which contains a list of marks that students received after going this path along with the number of occurrences (student cases for this mark). Additionally, weighted arithmetic average value (GAM) of these marks is represented in this label. The value of GAM serves as an estimation of success chances for future students that plan to go the same path.

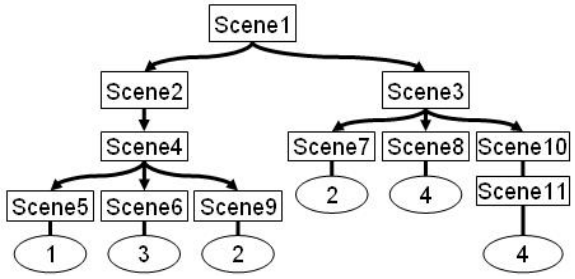


Fig. 5. A Simplified Decision Tree

Figure 5 shows a small example of a storyboard-tree. For simplicity, the labels (elliptic nodes) are reduced to the value of GAM. According to the form of the scenes in a storyboard the atomic attributes of the tree are placed in rectangles. The path through the storyboard-tree is defined by its directed edges. Only the connection to the label is non-directed, because it refers to the complete path. In [2] a prototypical implementation of the procedure is given. The decision tree to be built is represented by Prolog facts

- *tree(root, [<subtrees>])*

and the success labels that form the leaves of the tree are represented by Prolog facts

- *assess(node, <statistics>)*

The statistics-parameter of the assess-predicate are the values which are defined in the label.

Utilizing the decision tree for path estimation and completion

If a student submits a plan for an intended curriculum, which is already represented in the decision tree (as a path from its root to a node that is succeeded by a label node), the prediction is very easily estimated by presenting the content of this label.

In the other case, i.e. if a student submits a curriculum plan, which is not represented in the decision tree so far, the most similar sub-path in the decision tree will be identified. Similarity, in this context, refers to the number of subjects of starting sequences of all represented paths. In other words, those paths in the tree will be identified, which have the longest starting part in common with the submitted curriculum. The last node of this path forms the root of several sub-trees, which represent remaining paths, which are all different from the submitted remaining path. As the success chance estimation, all label nodes of the sub-trees are merged and their common weighted arithmetic average will be provided. To indicate the degree of similarity, the length of the starting sequence divided by the entire length of the submitted path will be presented.

Of course, in such a case, the student may be interested in suggestions to modify the submitted path in a way that the success chance reaches an optimum. Modifying, in this context, means the exchange of the rest-path, which is submitted, but not represented in the decision tree, by the most successful one that is represented in the tree. Here, the "most successful" alternative rest-path is the one with the best weighted arithmetic average value among the paths represented in the sub-trees starting at the last node, which have the tree and the submitted curriculum in common.

Based on this modification suggestion for the rest path along with the similarity degree between the submitted and proposed path, the student can make a decision on whether or not holding on the submitted curriculum or modifying if according to an optimization of the success estimation.

IV. Summary and Outlook

Storyboarding is a way of managing didactic knowledge for organizing learning experience. Our storyboarding concept leads beyond the limits of software engineering. All didactic forms may be included (1) collaborative work, (2) competitive work, (3) classical learning forms, even "playful learning" by involving game situations.

The general idea and objective (and vision) of this research can be outlined as follows:

- *Let's make explicit what we talk about!*
  The idea is (semi-) formally represent didactics.
- *Let's apply such representations in (our university) practice!*
  This way, also non-experts in didactics will become able to process a model of didactics. In particular, for university teaching this seems to be very useful, because university teachers (professors and tutors), who are usually excellent experts in their subject, but do not necessarily have the didactic skills to teach their subject.
- *Let's explore conditions that can be (formally) checked!*
  This includes the verification of logical anomalies such as consistency conditions, but also has the potential to check topical teaching knowledge like invariants, didactic principles, and so on.
- *Let's check the result of applying certain didactics in a case study!*
  By validating applied didactics based on the degree of success, we will be able to identify successful paths through storyboards and distinguish them from the less successful ones.
- *Let's learn from the validation results!*
  Based on the results of validation, we will be able to refine didactic knowledge towards incremental improvement by its (re-) validation. The cycle of ongoing validation and refinements bags the chance to incrementally (evolutionary) improve the didactics of teaching.
- *Let's derive successful didactic patterns!*
  This is a vision: Deriving didactic patterns inductively from successful and failing examples. The idea is to discover some general patterns that (a) good examples (storyboard paths that usually end up with a high level of success) have in common and (b) do not appear in bad examples (storyboard paths that end up with failure or bad results).
- *Let's utilize these patterns!*

Of course, such patterns are the right components, when designing new storyboards. Thus, some day we may be able to support didactics by a design tool that makes use of a pattern library.

Three storyboards (one at at a US-, a Japanese -, and a German University) are prototype examples so far. These examples indicate that the concept is very general and "many purpose".

Because of clarity and simplicity, everybody can become a storyboard author. No Software technological Knowledge is needed, no specialized (expensive) tool is needed.

By storyboarding, didactic design became explicit and subject to evaluation and quality assurance:

1) Structure tests for verification are developed as a method to discover logical anomalies in storyboards.
2) An inheritance concept has been developed as a means of logical (deductive) inference over this knowledge representation.
3) Based on a set of operations that ensures logical correctness of storyboards, we developed a web-based storyboard development environment for our storyboards.
4) As a first step towards inductive inference over this knowledge representation, we developed a method to estimate success chances of intended storyboard paths by applying data mining methods to paths that have been traversed formerly and their related degree of success.

   Additionally, this approach also suggests a supplement to a given curriculum that leads to an optimum with respect to the success chances.

The latter approach has been developed for practical use at a Japanese university (Tokyo Denki University), where students are requested in their first semester to compose an individual study plan, which meets all formal regulations as well as their individual needs, desires, talents, opportunities, and carrier goals.

Since this approach is just prototypically implemented in Prolog, a next step towards it's utilization is designing an interface between the SQL-database behind the storyboard development environment in [14] and the Prolog implementation of the approach of [2].

In fact, the above mentioned list of objectives and visions starts with items that are well done so far, but ends up with items that are hard to achieve and subject of much research left. In particular, the last two items are not touched at all so far, but they are our dream and ultimate goal.

REFERENCES

[1] Balaji, V., Arora, A., Jain, A., Goyal, G., Dubey, G., and Singh, S., SCORM Learning Sequence Modeling with Petri Nets in Cooperative Learning, *Learning Technology*, 7(1), Special Issue on SCORM 2004 Sequencing & Navigation, ISSN 1438-0625, 2005.

[2] Boeck, R., *Ein Data Mining Verfahren zur Pfadbewertung in Storyboards* (German), Diploma Thesis, University of Ilmenau, Faculty of Computer Science and Automation, Chair of Artificial Intelligence, 2007.

[3] Bransford, J.D., Brown, A.L., and Cocking, R.R., *How People Learn: Brain, Mind, Experience, and School*, National Academic Press, 2000.

[4] Briggs, R.M., Gagne, L.J., and Wager, W.W., *Principles of Instructional Design*, Thomson Learning, 1992.

[5] Damasio, A., *The Feeling of What Happens: body and emotion in the making of consciousness*, Hartcourt, 1999.

[6] Davis, B., Sumara, D., and Luce-Kapler, R., *Learning and Teaching in a Complex World*, Lawrence Erlbaum Associates, 2000.

[7] Dohi, S., Nakamura, Y., Sakurai, Y., Tsuruta, S., and Knauf, R., Dynamic Learning Need Reflection System for academic education and its applicability to Intelligent Agents, *Proc. of the 6th IEEE International Conference on Advanced Learning Technologies (ICALT 2006)*, Kerkrade, the Netherlands, pp. 459-463, ISBN 0-7695-2632-2,Los Alamitos, CA: IEEE Computer Society, 2006.

[8] Dohi, S., Sakurai, Y., Tsuruta, S., and Knauf, R., Managing academic education through dynamic storyboarding, *Proc. of the World Conference on e-Learning in Corporate, Government, Healthcare,and Higher Education 2006 (E-Learn 2006)*, Honolulu, Hawaii, USA, Chesapeake, VA: Association for the Advancement of Computing in Education (AACE), ISBN: 1-880094-60-6, pp. 1611-1619, 2006.

[9] Duesel, H., *Konzeption und Realisierung von Methoden der formalen Verifikation von Storyboards* (German), Diploma Thesis, University of Ilmenau, Faculty of Economic Sciences, 2007.

[10] Feyer, T. and Thalheim, B., E/R based scenario modelling for rapid prototyping of web information services, P.P.-S. Chen (ed): *Advances in Conceptual Modeling*, vol. 1727 of *Lecture Notes in Computer Science (LNCS)*, pp. 253-263, Berlin: Springer, 1999.

[11] Flechsig, K.-H., *Kleines Handbuch didaktischer Modelle* (German), Neuland, 1996.

[12] Forsha, H.I., *The Complete Guide to Storyboarding and Problem Solving*, ASQ Quality Press, 1994.

[13] Jantke, K.P. and Knauf, R., Didactic design though storyboarding: Standard concepts for standard tools, *Proc. of the 4th Internat. Symposium on Information and Communication Technologies, Workshop on Dissemination of e-Learning Technologies and Applications*, Cape Town, South Africa, 2005, pp. 20-25, New York: ACM Press, ISBN 0-9544145-6-X, 2005.

[14] Kasperski, S., *Entwicklung eine unabhaengigen Storyboardrepraesentation* (German), Studienarbeit, University of Ilmenau, Faculty of Computer Science and Automation, Chair of Artificial Intelligence, 2007.

[15] R. Knauf, Y. Sakurai, S. Tsuruta, Toward Making Didactics a Subject of Knowledge Engineering, *The 7th IEEE International Conference on Advanced Learning Technologies* (ICALT 2007), Niigata (Japan), Las Alamitos, CA: IEEE Computer Society, ISBN 0-7695-2916-X, pp. 788-792, 2007.

[16] Lin, J., Ho, C., Sadiq, W., Orlowska, W.E., Using Workflow Technology to Manage Flexible e-Learning Services, *Educational Technology and Society*, 5(4), 2002.

[17] Schewe, K.-D. and Thalheim, B., The Co-Design Approach to WIS Development in E-Business and E-Learning Applications, *Proc. of FIPWIS 2004*, November 24, Brisbane, Australia, 2004.

[18] Rothwell, W.J. and and Kazanas, H.C., *Mastering the Instructional Design Process: A Systematic Approach*, Third Edition Pfeiffer, 2004.

[19] Sauerstein, G., *Formale Modellierung und daduktive Interpretation von Storyboards. Modellierung eines komplexen Prozessmodells* (German), Studienarbeit, University of Ilmenau, Faculty of Computer Science and Automation, Chair of Artificial Intelligence, 2006.

[20] Sykes, E.R. and Franek, F., A prototype for an intelligent tutoring system for students learning to program in Java, *Proc. of the IASTED International Conference on Computers and Advanced Technology in Education*, Rhodes, Greece, pp. 78-83, 2003.

[21] Xu, G., *Ein Vererbungskonzept fuer Storyboards* (German), Studienarbeit, University of Ilmenau, Faculty of Computer Science and Automation, Chair of Artificial Intelligence, 2006.